

UNIVERSITI TEKNOLOGI MALAYSIA**DECLARATION OF THESIS / UNDERGRADUATE PROJECT PAPER**

Author's full name : MOHAMAD KHAIRUDIN FAIZ BIN MANSOR

Date of Birth : 24 AUGUST 1992

Title : KINECT BASED TABLE-TENNIS TRAINER SYSTEM

Academic Session : **2014/2015**

I declare that this final year project report is classified as:

☐

CONFIDENTIAL

(Contains confidential information under the Official Secret Act 1972)*

☐

RESTRICTED

(Contains restricted information as specified by the organization where research was done)*

☒

OPEN ACCESS

I agree that my final year project report to be published as online open access (full text)

I acknowledged that Universiti Teknologi Malaysia reserves the right as follows:

1. The thesis is the property of Universiti Teknologi Malaysia
2. The Library of Universiti Teknologi Malaysia has the right to make copies for the academic purposes.

Certified by:

SIGNATURE

920824-06-5969
(NEW IC NO/PASSPORT)

Date: 28th JUNE 2015

SIGNATURE OF SUPERVISOR

DR ABDUL RASHID BIN HUSAIN
NAME OF SUPERVISOR

Date: 28th JUNE 2015

NOTES: * If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter from the organization with period and reasons for confidentiality or restriction.

“I hereby declare that I have read this final year project report and in my opinion this final year project report is sufficient in terms of scope and quality for the award of the degree of Bachelor of Engineering (Electrical - Mechatronics)”

Signature :
Name of Supervisor : DR. ABDUL RASHID BIN HUSAIN
Date : 28th JUNE 2015

KINECT BASED TABLE – TENNIS TRAINER SYSTEM

MOHAMAD KHAIRUDIN FAIZ BIN MANSOR

A final year project report submitted in partial fulfillment of the
requirements for the award of the degree of
Bachelor of Engineering (Electrical - Mechatronics)

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JUNE 2015

I declare that this final year project report entitled “KINECT BASED TABLE – TENNIS TRAINER SYSTEM” is the result of my own research except as cited in the references. The final year project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature : _____
Name : MOHAMAD KHAIRUDIN FAIZ BIN MANSOR
Date : 28th JUNE 2015

*To my lovely mother, **Zarina Anuar** who gave me endless love, trust, constant encouragement over the years, and for her prayers.*

*To my **Family**, for their patience, support, love, and for enduring the ups and downs during the completion of this thesis.*

*To my **Friends**, for their guidance during the completion of this thesis.*

This report is dedicated to them.

ACKNOWLEDGEMENT

I wish to express my deepest appreciation to all those who helped me, in one way or another, to complete this project. First and foremost I thank God almighty who provided me with strength, direction and purpose throughout the project. Special thanks to my project supervisor Dr. Abdul Rashid Husain for all of his patience, guidance and support during the execution of this project. Through his expert guidance and help, I was able to overcome all the obstacles that I encountered in these enduring ten months of my project. In fact, he always gave me massive advice every time I consulted with him over problems relating to my project. Last but not least I would like to express my gratitude to all of my friends who helping me during this tough period of this project. Without guidance from them I may not be able to complete this project.

ABSTRACT

Kinect or previously known as Project Natal is a line of motion sensing input devices developed by Microsoft to enable user to interact or control certain applications or games in Xbox 360 or Xbox One. Kinect can be used as a replacement for the traditional controller that has been used widely in console game. The first generation of Kinect has been introduced in November 2010 and later on Microsoft launched Kinect for Windows on February 1, 2012. Microsoft in fact realized that there have a lot of people using Kinect not only in Xbox 360 application only but also in other application based on Windows platform. Mostly people use Kinect in image processing but it is hard to find other applications which combine Kinect Skeletal Tracking with simple and affordable microcontroller such as Arduino Uno. Kinect based Table – Tennis Trainer (3T) System is designed to enhance the existing Table – Tennis trainer robot developed by previous developer by implementing Kinect and also low cost microcontroller such Arduino Uno. 3T is a table – tennis ball launcher for practice purpose and with this Kinect system, player can control the direction of the ball and also the type of the spin produced by this launcher. By controlling the angle of servo motor, the direction of the ball and type of spin can be selected by user using only hand gesture. In conclusion, this system can enhance the functionality of the Table – Tennis Trainer (3T) without dropping its performance.

ABSTRAK

Kinect ataupun sebelum ini diberi nama Projek Natal merupakan peranti pengesanan input gerakan yang dibangunkan oleh Microsoft untuk membolehkan pengguna berinteraksi atau mengawal aplikasi dan permainan video tertentu di dalam Xbox 360 dan Xbox One. Kinect juga boleh dianggap sebagai alternatif kepada pengawal generasi lama yang sering digunakan di dalam permainan konsol. Generasi pertama Kinect telah dilancarkan pada November 2010 dan selang beberapa tahun kemudian Kinect khas untuk Windows telah dilancarkan oleh Microsoft pada 1 Februari 2012. Microsoft menyedari terdapat beberapa pengguna yang menggunakan Kinect bukan sekadar untuk kegunaan di Xbox 360 sahaja akan tetapi untuk aplikasi lain yang menggunakan platform berdasarkan Windows. Kebanyakan pengguna menggunakan Kinect untuk pemprosesan imej tetapi sukar untuk melihat aplikasi yang menggabungkan penggunaan sistem Kinect Skeletal Tracking bersama dengan mikropengawal yang berpatutan seperti Arduino Uno. Sistem Pelatih Ping Pong (3T) berdasarkan Kinect telah direka untuk meningkatkan keupayaan Pelatih Ping Pong (3T) yang telah dibangunkan oleh pelajar UTM dengan mengaplikasikan penggunaan Kinect dan juga mikropengawal kos rendah seperti Arduino Uno. 3T merupakan pelancar bola ping – pong untuk latihan dan dengan kehadiran system Kinect ini, pengguna boleh mengawal arah bola dan juga jenis putaran yang boleh dijana oleh 3T. Pengguna boleh memilih jenis putaran bola dan juga arah bola yang dikehendaki dengan mengawal nilai sudut untuk motor servo melalui isyarat tangan sahaja. Kesimpulannya system ini dapat meningkatkan fungsi yang terdapat di dalam 3T tanpa menurunkan prestasi 3T itu sendiri.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	DECLARATION	ii
	DEDICATION	iii
	ACKNOWLEDGEMENT	iv
	ABSTRACT	v
	ABSTRAK	vi
	TABLE OF CONTENTS	vii
	LIST OF TABLES	x
	LIST OF FIGURES	xi
	LIST OF ABBREVIATIONS	xiii
1	INTRODUCTION	1
	1.1 Project Background	1
	1.2 Problem Statement	2
	1.3 Objective	2
	1.4 Scope of Project	3
	1.5 Project Outline	3
2	LITERATURE REVIEW	4
	2.1 Introduction	4
	2.2 Kinect	4
	2.3 Kinect Application for Windows	5
	2.4 Kinect Application for Non – Windows	6
	2.5 Components and Tools	6
	2.5.1 Microcontroller	7
	2.5.2 AC to DC Converter	8

2.6	Technique	10
2.6.1	Depth Data	10
2.6.2	Skeletal Tracking	11
2.7	Summary	11
3	METHODOLOGY AND DEVELOPMENT OF KINECT BASED TABLE TENNIS TRAINER SYSTEM	12
3.1	Introduction	12
3.2	Project Flow	13
3.3	Kinect for Windows SDK	14
3.4	Circuit Design and Construction	15
3.5	Microcontroller Module	16
3.6	Motor Driver	17
3.7	Power Supply	18
3.8	Robot Programming	18
3.8.1	Flow Chart	19
3.8.2	Arduino IDE Programming	20
3.8.3	Motor Control	20
3.8.4	Microsoft Visual Studio 2012 Programming	22
3.9	Summary	26
4	RESULTS AND DISSCUSSIONS	27
4.1	Introduction	27
4.2	Power Supply Circuit	27
4.3	Kinect Graphical User Interface	29
4.4	System Performance	29
4.5	Summary	33
5	PROJECT MANAGEMENT	34

5.1	Introduction	34
5.2	Project Schedule	35
5.3	Cost Estimation	36
5.4	Summary	37
6	CONCLUSIONS AND RECOMMENDATIONS	38
6.1	Introduction	38
6.2	Conclusion	38
6.3	Recommendations on Future Work	39
	REFERENCES	40
	APPENDIX A	43
	APPENDIX B	44

LIST OF TABLES

TABLE NO.	TITLE	PAGE
2-1	Comparison of Kinect research field among researcher	6
2-2	Comparison of commercialized microcontroller for non – industrial purpose	7
2-3	Comparison of Three Different AC to DC Converter Device in Market	9
3-1	Comparison of different type of motor available in 3T	18
5-1	Cost Estimation for Electrical Hardware	37

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
2.1	All of the main components for Kinect	5
2.2	Arduino Uno	8
2.3	STM32VLDISCOVERY	8
2.4	AC to DC Adapter 12V 2A	9
2.5	50W Single Output Switching Power Supply	9
3.1	Block diagram of table tennis Trainer (3T) system with Kinect	13
3.2	Process flow of project	14
3.3	The control circuit	15
3.4	Arduino Uno microcontroller board	16
3.5	Motor Driver 2A shield	17
3.6	MD10 Shield	17
3.7	Flow chart for Kinect Based Table Tennis Trainer System	19
3.8	Global declaration for two servos.	21
3.9	Functions for servo control in Arduino IDE	21
3.10	Initialization of pin for DC motor	22
3.11	Setting up the entire pin for motor driver shield to become output	22
3.12	Enabling Kinect in Visual Studio 2012	23
3.13	Scaling value X and Y axis for all Joint captured by Kinect.	23
3.14	Setting serial communication in Visual Studio 2012	24
3.15	Coding section for requesting predefined code for each joint.	25

3.16	Function to send X value, Y value and joint code to Arduino Uno board.	25
4.1	Layout of power supply circuit.	28
4.2	Switching power supply 12V 4.5A	28
4.3	Graphical User Interface for Kinect based Table Tennis Trainer System	29
5.1	Project Management – Time, Cost, quality triangle	34
5.2	Project Gantt Chart (semester one)	35
5.3	Project Gantt Chart (semester two)	36

LIST OF ABBREVIATIONS

3T	-	Table Tennis Trainer
DC	-	Direct Current
AC	-	Alternate Current
FYP	-	Final Year Project
I/O	-	Input/output
SDK	-	Software Development Kit
PWM	-	Pulse Width Modulation
GUI	-	Graphical User Interface
NA	-	Not Available
USB	-	Universal Serial Bus

CHAPTER 1

INTRODUCTION

1.1 Project Background

Microsoft announced Kinect on June 1, 2009 under the codename Project Natal named after Brazilian city of Natal. Natal is birthplace of Brazilian – born Microsoft director Alex Kipman. Until February 2013 Microsoft managed to ship 24 million units of Kinect and Microsoft also having sold 8 million units of Kinect in first 60 days on the market.

Even though Kinect has been used mainly for Xbox 360 or Xbox One purpose, there have some people outside there who develop other application which can utilize Kinect outside Xbox 360 or Xbox One environment. The other environment is Windows platform. In March 2012 Microsoft reported that almost 350 companies are working with them to develop Kinect applications for windows platform. The prospect of Kinect to control 3T robot developed by Lee Poh Yong exist due to fact that Kinect can be run on Windows platform and also 3T does not has any remote controller system. Usually most of the table – tennis robot available in market equipped with remote controller for user to control the robot.

With Kinect Skeletal Tracking method, user can use only hand gesture and motion to control 3T without using any physical remote controller. Kinect has range of detection between 1.8 meter for single player and 2.5 meter for two players which can be considered adequate for Table Tennis Trainer application. This system also can replace the existing GUI which need player to set the direction and type of ball spin at computer first before training session begin.

1.2 Problem Statement

Table - Tennis Trainer or 3T is a robot for training purpose developed by Lee Poh Yong with graphical user interface or GUI that can enable user to select different drill for different level of table – tennis training session. Even though the GUI for this robot is quiet good and easy for use, but the robot is lacking in term of remote controlling system and also stable power source to run all of the DC motor and servo motor available. Previously 3T also used a microcontroller that is not available in market due to the limitation of distribution by National Instrument. With implementation Kinect, Arduino Uno microcontroller and also a stable power supply system, an efficient remote controller system can be designed on 3T.

1.3 Objective

- 1) To design a remote controller system based on Kinect to control 3T robot efficiently.
- 2) To operate 3T robot with lower cost microcontroller as compare as previous one.
- 3) To design a stable and efficient power supply circuit for 3T robot.

1.4 Scope of Project

- 1) Kinect system that able to allow user to control direction and type of ball spin produced by 3T robot.
- 2) Microcontroller that is cheaper and reliable compare to previous microcontroller.
- 3) Stable and efficient power supply system for running 3T robot.

1.5 Project Outline

This report consists of six chapters which discuss about the project background, objective and scope of this project. Chapter 2 presents literature review about the project that includes history of Kinect, existing Kinect application, component to be used and project by other previous researcher. Chapter 3 explains the methodology and development Kinect based 3T system. This chapter is divided into three main parts which are power supply circuit design, Arduino Uno programming and Kinect programming. On the other hand, Chapter 4 presents the result of project and discussion regarding the result. Chapter 5 discusses the project management which includes Gantt chart and cost estimation of project. Lastly, Chapter 6 concludes the overall of project and provides recommendation for future works.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In order to be better understanding on the project to be implemented, literature review and study have been conducted towards different type of Kinect application, project component, tools and technique that used by other researcher.

2.2 Kinect

Kinect has five main parts which are depth sensing system, RGB camera, accelerometer, and tilt motor and lastly microphones. All of this part play important role for Kinect in order to function properly in Xbox 360 and Xbox One applications or games. Kinect have two options for the RGB camera which can run at 30fps at 640x480 pixels or 10fps at 1280x1024 pixels [1].

The RGB camera itself has several features which are automatic white balancing, black reference, and flicker avoidance, color saturation and defect correction. Next for the depth sensing system it consists of the IR laser emitter an IR camera separately from the RGB camera. In order to measure the depth, the depth

sensing uses structured light method and the IR laser emitter will project a known pattern of dots [1].

The tilt motor has some gearing to enable the Kinect to tilt up and down. For the accelerometer, it has a function to enable the system to calibrate to a value that can allow the head to be moved at specified angles. Kinect also equipped with 4 microphone capsules with 16 bit- audio processing capability [1]. Figure 2.1 shows all important parts for Kinect to work as gesture motion sensor.

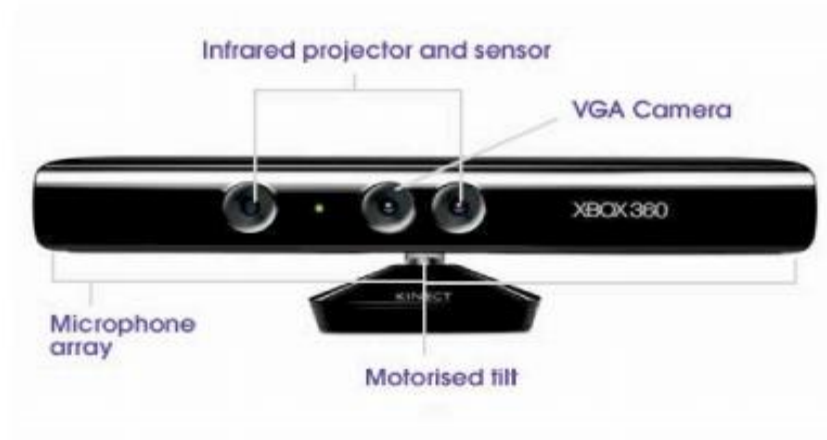


Figure 2.1 All of the main components for Kinect

2.3 Kinect Application for Windows

Nowadays more than 30 applications have been developed by developers around the world for Windows 8 application. All of the applications available at Windows Store mostly based on Kinect for Windows sensor and has several category including health, education, photo and many more. Since Microsoft launched Kinect for windows SDK, there have a lot of developer outside there try to develop a good and interesting application that can be downloaded by consumer at Windows Store.

2.4 Kinect Application for Non – Windows

Other than commercial applications that targeting ordinary consumer, there has also some researcher that use Kinect in more advance application such as in embedded system, in healthcare system and many more. Most researcher use Kinect in image processing application involving smart house application, medical checkup application and also 3D mapping. Some example of research involving Kinect technology was shown in Table 2.1

Table 2-1 Comparison of Kinect research field among researcher

Title of Research	Face Recognition System Based On A Kinect Sensor and Windows Azure Cloud Technology [2]	Patient Monitoring during External Beam Radiotherapy[3]	Study on the Use of Microsoft Kinect for Robotics Applications
Type of Research	Security	Medical	Machine Vision
Kinect Feature	Skeletal Tracking	Depth Sensor	Depth sensor
Language of Programming	C#	C#	C#
Other Feature	Neural Network	Matlab	Inertial Measurement Unit

2.5 Components and Tools

Components and tools must be choose widely especially for microcontroller and power supply part. 3T does not have electrical hardware and also microcontroller anymore and it cannot run properly. Hence a power distributor circuit and a microcontroller system need to be built in order to run 3T and integrate it with Kinect.

2.5.1 Microcontroller

Microcontroller can be defining as a small computer on a single integrated circuit. It contains a processor core, memory and programmable input/output peripherals. Microcontroller has been used widely in various fields such as automation, automobile, medical and other embedded systems. The first microcontroller has been produced by Texas Instrument engineers Gary Boone and Michael Cochran in 1971. The microcontroller has been named TMS 1000 which become commercially available in 1971 [7]. Nowadays there have a lot of microcontroller available in market with various specifications either specialized for industry purpose or for casual use. The price also depends on the specification and the ability of the microcontroller to deliver better performance during heavy utilization. Some example and comparison of commercialized microcontroller for non – industrial purpose was shown in Table 2.2. Figure 2.2 and 2.3 shows them difference between Arduino Uno board and STM32VLDISCOVERY board in term of size and components configuration.

Table 2-2 Comparison of commercialized microcontroller for non – industrial purpose

	Arduino Uno	MSP-EXP430G2 (Launchpad)[8]	Kit Pinguino 32MX250	STM32VL DISCOVERY
Microcontroller	ATmega 328	MSP430G2553	MIPS M4K core	ARM 32-bit Cortex™-M3 CPU
Operating Voltage	5V	3.6V	4V	5V/3V
Digital I/O Pins	14	20	19	80
Flash Memory	32KB	16kB	128KB	128KB
SRAM	2KB	512B	32KB	8KB
Clock Speed	16MHz	16MHz	40MHZ	24MHz



Figure 2.2 Arduino Uno



Figure 2.3 STM32VLDISCOVERY

2.5.2 AC to DC Converter

AC to DC converter is a device that can convert alternate current (AC) that come from power plug or socket at home into direct current (DC). 3T robot mainly used DC motor and servo motor which required DC power supply. There has a lot of type AC to DC converter such as rectifier, mains power supply unit (PSU) which has been used in desktop or personal computer application and also switched - mode power supply. Table 2-3 shows the comparison between different types of AC to DC converter in current market.

Table 2-3 Comparison of Three Different AC to DC Converter Device in Market

	AC – DC Converter 12V	50W Dual Output Switching Power Supply[5]	50W Single Output Switching Power Supply[6]
Model	DSA-0151F-12A	D – 50A	NES-50-12
Input Voltage	100-240V at 50/60Hz	85 – 132 VAC 170 – 264 VAC Selection by switch	85-264VAC
Input Current	0.4A	1A/115VAC 0.52A/230VAC	1.1A/115VAC 0.65A/230VAC
Output Voltage	+12V DC	5VDC 12VDC	12VDC
Output Current	2A	6A 2A	4.2A



Figure 2.4 AC to DC Adapter 12V 2A



Figure 2.5 50W Single Output Switching Power Supply

Figure 2.4 and 2.5 shows the difference between two types of AC and DC converter in term of size. AC to DC Adapter 12V 2A has smaller and more compact size compare to 50W Single Output Switching Power Supply.

2.6 Technique

In this section, we will study two type of method for Kinect that has been used by previous researcher for their research.

2.6.1 Depth Data

Depth data use a measurement of depth which has been describe by the inventors as triangulation process. A diffraction grating will split a single beam from laser emitter source into multiple beams to create a constant pattern of speckles projected onto the scene. This is the actual pattern that later on will be compared with reference pattern. The sensor later on will capture a plane at a known distance to obtain the reference pattern and then the reference pattern will be saved into the memory of the sensor. When a speckle is projected on an object whose distance to the sensor is different than that of the reference plane the position of the speckle in the infrared image will be shifted in the direction of the baseline between the perspective centre of the infrared camera and the laser projector. A simple image correlation procedure will measure every single shift for all speckles which later on will yields a disparity image [9].

2.6.2 Skeletal Tracking

Skeletal tracking is a feature that brings Kinect lives up with reputation as the next generation of motion sensor. Skeletal tracking utilize depth data to estimate the positions of twenty predefined body joints that constitute a wireframe skeleton of a user that moving in the field of view of Kinect [10]. It allows developer or researcher to turn human body works as a controller for any system that required remote controller to make it function.

2.7 Summary

As summary for this chapter, various type of research related with Kinect has been conducted outside there. Besides that, two type of Kinect features that has been used by previous researcher also been study to understand more about Kinect.

CHAPTER 3

METHODOLOGY AND DEVELOPMENT OF KINECT BASED TABLE TENNIS TRAINER SYSTEM

3.1 Introduction

This project with title Kinect Based Table-Tennis Trainer System targeted to improve and enhance user experience for the available product which is Table Tennis Trainer (3T) by providing a remote controller system based on Kinect. The Table tennis Trainer or 3T was developed by past undergraduate student Lee Poh Yong. As shown in Figure 3.1, Arduino Uno was selected as main controller for 3T system to replace the previous microcontroller which is NIsbRIO-9642XT. While the entire actuator robot such as DC permanent magnet motor and positional rotation servo still in good conditions.

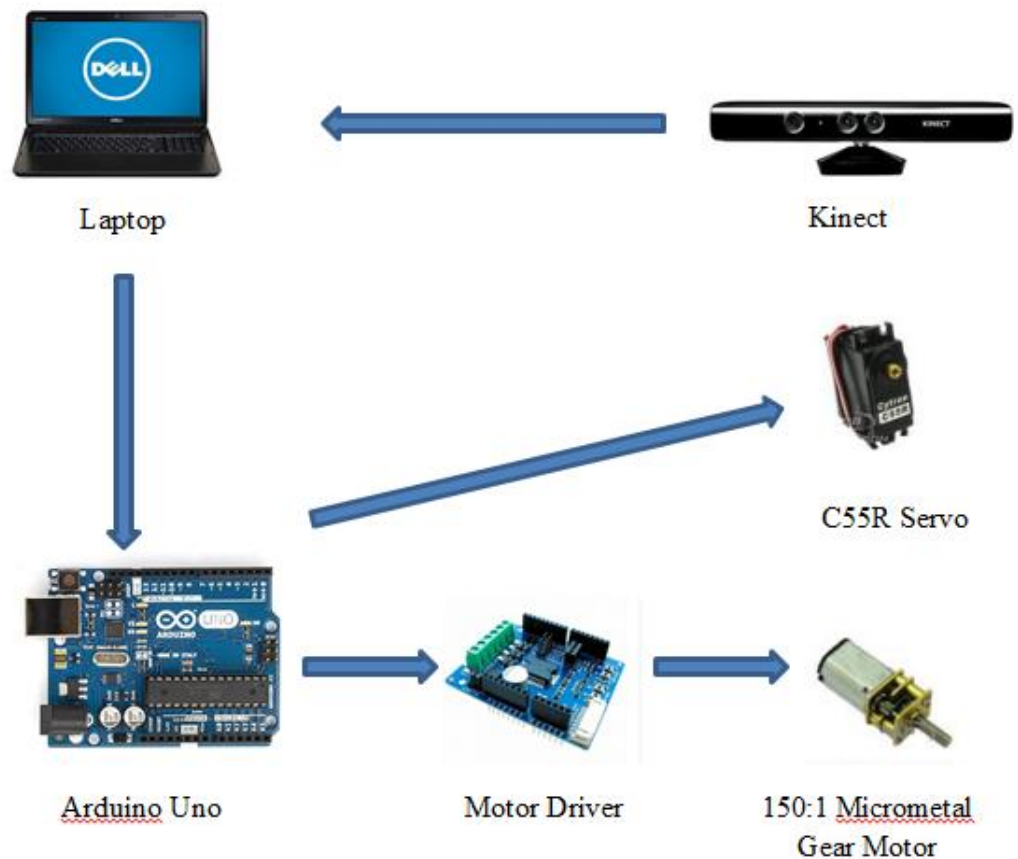


Figure 3.1 Block diagram of table tennis Trainer (3T) system with Kinect

As shown in figure 3.1 Kinect required a laptop or personal computer to process the data before the data will be sent to Arduino Uno to control all of the available motors.

3.2 Project Flow

This project was divided into seven phases for this FYP 2. Mostly all of the phases are including identification problem, literature review, circuit design, buying tools and equipment, building Kinect system, troubleshooting, documentation and reporting. Figure 3.2 shows the process flow of project which start with problem identification and end with documentation and reporting.

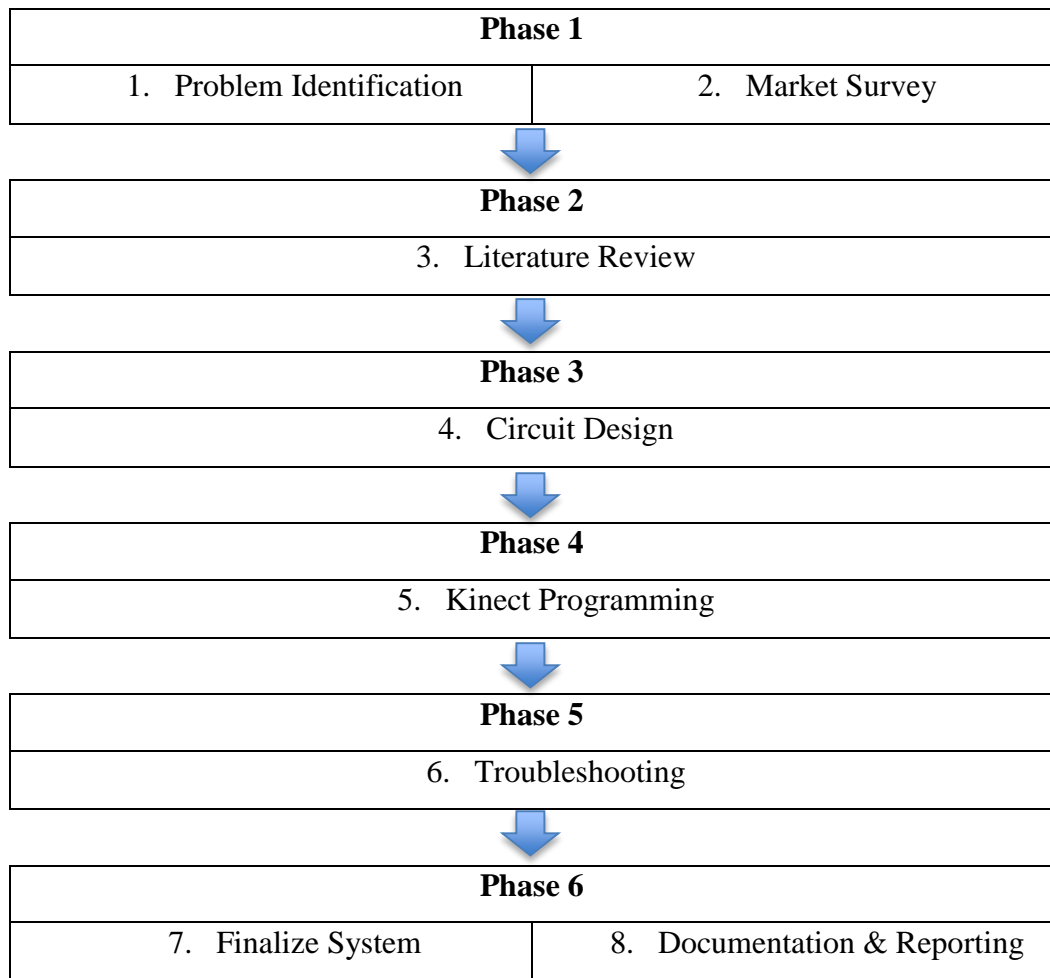


Figure 3.2 Process flow of project

3.3 Kinect for Windows SDK

Kinect for Windows Software Development Kit or SDK has been released by Microsoft for all of the developer outside there who want to use and develop Kinect apps for running in Windows operating system. This SDK has several versions started from beta until version 2.0 which released by Microsoft for Windows 8.1 user and also for Kinect for Windows Sensor compatibility. This SDK will enable Microsoft Visual Studio to insert plugin Kinect so all of the coding syntax for Kinect can be enable in Microsoft Visual Studio coding environment. For this project Kinect for Windows SDK version 1.8 has been used since it support for Windows 7

operating system which are the operating system for the project laptop while for the version 2.0 it just support for only Windows 8.1 and Windows 8.

3.4 Circuit Design and Construction

The main components of controller circuit used for this project were microcontroller module and motor driver shield. All of these components have purpose to control the DC permanent magnet motor and positional rotation servo which were used as the actuators for the robot. Pulse width modulation (PWM) generated by Arduino Uno through its digital I/O was used to control the speed of DC motor or angle of rotation of servo motor. Furthermore, for this project we will use Kinect to control the value of Pulse Width Modulation or PWM given by Arduino using Kinect by reading gesture from user. The motor driver shield MD10 and 2A motor driver shield also were used to drive the DC motor which will perform ball shooting and regulate the frequency of the output shooting. Servo motor can be directly controlled by Arduino.

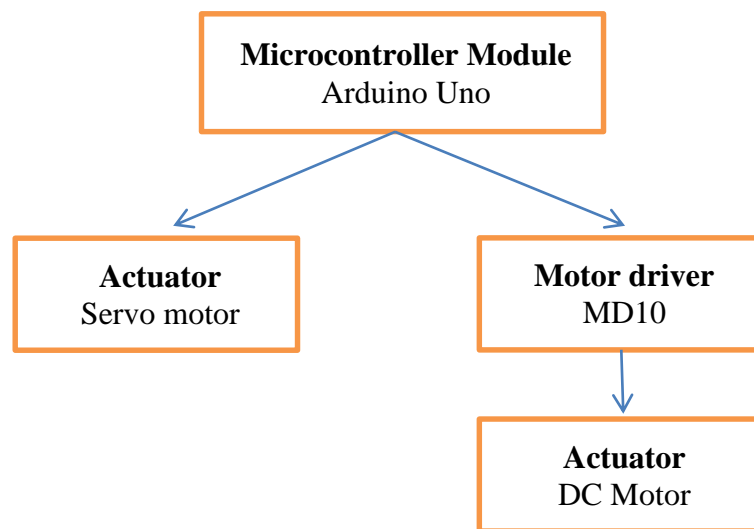


Figure 3.3 The control circuit

3.5 Microcontroller Module

The previous microcontroller module selected for controlling the Table tennis Trainer was NI sbRION-9624XT. This microcontroller used by previous student Lee Poh Yong since he was entering Innovate Malaysia competition under National Instrument track. After the competition end, he need to return back the microcontroller to National Instrument and the robot is left without any microcontroller. After surveying several potential microcontrollers to be used in this project, Arduino Uno as shown in figure 3.4 has been selected. Arduino Uno comes with several features:

- Microcontroller used is ATmega328
- 5V operating voltage
- 14 digital I/O pins
- 6 analog input pins
- 16 MHz clock speed
- 2KB SRAM



Figure 3.4 Arduino Uno microcontroller board

Even though the specifications and features is less powerful compare to the previous microcontroller that has been used, but it is enough to drive this robot and make it work at optimum performance. It is also cheap since the price is only RM55.

3.6 Motor Driver

The motor driver which is used to drive the robot actuator is a shield which can be attached directly onto the Arduino Uno. There have two types of motor driver used in this project which are shield MD10 and shield 2A. The only differences between this two shield is the maximum current that can be driven by MD10 is 10 Ampere [11] while for shield 2A is only 2Ampere [12]. Both of them will be used to drive 2 DCmotor with 3 volt rating and one Micrometal Gear Motor with 5 volt.

As we can see in figure 3.5 and figure 3.6, the differences of these two models is the number of channel available with MD10 shield has one channel more compare to Motor Driver 2A Shield.



Figure 3.5 Motor Driver 2A shield



Figure 3.6 MD10 Shield

3.7 Power Supply

The other part for this project is power supply which is important since we have two servo motor and three DC motor. In order to supply voltage efficiently to all of the available motor we need to determine first the voltage requirement for all of the motor. Table 3.1 shows the type of motor and the specifications for each of them.

Table 3-1 Comparison of different type of motor available in 3T

Motor	Voltage
3V miniature brush motor	3V
150:1 micro metal gearmotor [13]	3-9V
Servo Motor(C40R) [14]	4.8V/6.0V
Servo Motor(C55R)[14]	4.8V/6.0V

From this table the suitable power supply is 50W switching power supply with ability to supply up to 12V and 4.5A. This switching power supply can convert AC power from socket into DC power. The ability to deliver high current also can ensure that this power supply able to deliver efficient current to 3T robot without dropping its performance. Since this power supply deliver a fixed voltage of 12v, voltage regulator need to be used in this system so we can use the power at the required value given by the motors specification. Since there have only 3V and also 6V we need to use 2 type of voltage regulator which are voltage regulator 3.3V and voltage regulator 6V available at Cytron.

3.8 Robot Programming

In order to operate a robot with Kinect controller system, programming is important to declare the input and output, construct algorithm, and controlling the I/O so that desired operation can be perform.

3.8.1 Flow Chart

Figure 3.7 shows the flow chart of Kinect system for 3T. user will first choose their direction of the ball output and type of spin, the message will be process and convert into PWM to control the DC motor and also servo motor.

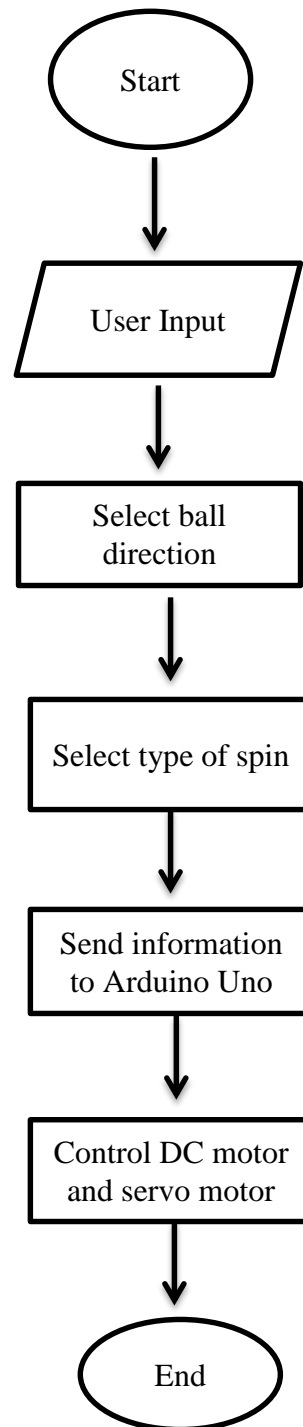


Figure 3.7 Flow chart for Kinect Based Table Tennis Trainer System

3.8.2 Arduino IDE Programming

Arduino IDE is software from Arduino which can allow developer to run and debug any program or code for any line up product from Arduino. It is a free software aimed to encourage people to try Arduino while enjoy doing mini project or any research by using Arduino product. This Arduino IDE use C or C++ language as the main language for programming [15].

The reason Arduino IDE is choosen because of Arduino Uno using this IDE as firmware and also Arduino IDE has a set of syntax or code that easy to understand by a beginner that try to use microcontroller for the first time. Even though it performance is not same like NI sbRIO-9642XT that has been used previously but for this project it still adequate to run 3T and receive any data from PC or laptop that connected with it alongside Kinect. Servo Motor will be controlled by how many input received from Microsoft Visual Studio 2012 through serial communication. While for DC motor the value of PWM will be set in Arduino IDE.

3.8.3 Motor Control

This project has been built with two servo motor which are positional rotation servo. Supposedly the angle for the servo motor to turn should be determined by the duration of a pulse that is applied to the signal wire or PWM. But Arduino IDE already simplified it and made it easy by providing user a library which can control servo motor by giving only degree of the motor to turn without using PWM method. Figures 3.8 shows how the library can be used to control the servo motor.

```
#include <Servo.h>

int MotorDC_PWMR = 5;    //initialization of pin for DC motor
int MotorDC_PWML = 6;
int MotorDC_PWMER = 3;

Servo yaw servo;          //global declaration for two servo
Servo pitch servo;
```

Figure 3.8 Global declaration for two servos.

After the declaration has been made, the library can be used and in order to control the servo motor Arduino provide several function exclusive for servo position control. In this project two functions has been used which are attach() and write(). Function attach() used to assign pin on Arduino board that has been attached to servo motor. Meanwhile function write() enable developer to give angle value for the servo in term of degree such as 90°. Figures shows all the functions used to control servo motor.

```
yaw servo.attach(9);      //initialization pin for yaw servo
pitch servo.attach(8);    //initialization pin for pitch servo
yaw servo.write(90);      //default position
pitch servo.write(0);     //default position
```

Figure 3.9 Functions for servo control in Arduino IDE

Controlling DC motor does not require library. By manipulate the PWM or duty cycle supplied to Arduino motor driver shield, required voltage can be obtained and supply to DC motor. Arduino motor driver shield that has been used in this project has provided all of the I/O pin assignment for controlling direction and speed of DC motor attached on it. In other meaning developer cannot choose their own I/O pin for controlling DC motor since the shield already assigned the required pins and it already fixed. For MD10 motor driver shield, pin 3 was set for PWM and pin 2 for setting direction either clockwise or vice versa. Since Motor driver shield 2A has two channels so there have four I/O pins already set up for this shield. I/O pin 5 and 6 has been set for PWM value of motor 1 and motor 2 respectively while pin 4 and 7 for the direction of motor 1 and motor 2 respectively. Figures 3.10 and 3.11 show how the coding is used in Arduino IDE.

```
#include <Servo.h>

int MotorDC_PWMP = 5;    //initialization of pin for DC motor
int MotorDC_PWML = 6;
int MotorDC_PWMBR = 3;
```

Figure 3.10 Initialization of pin for DC motor

```
//declaration for all of the pin for DC motor part
pinMode(MotorDC_PWMBR,OUTPUT);
pinMode(MotorDC_PWML,OUTPUT);
pinMode(MotorDC_PWMP,OUTPUT);
pinMode(7,OUTPUT);
pinMode(4,OUTPUT);
pinMode(2,OUTPUT);
pinMode(9,OUTPUT);
```

Figure 3.11 Setting up the entire pin for motor driver shield to become output

Figure 3.11 explains that in order to enable all of the assigned pin, function `pinMode()` need to be set as output so the Arduino can supply PWM value digital value to direction and PWM pins for motor driver shield.

3.8.4 Microsoft Visual Studio 2012 Programming

There has a lot of ways to programming Kinect using Windows 7 as operating System. Microsoft also release Kinect for Windows SDK to allow developer to build applications for Kinect based on Windows operating system or for their research purpose. One of the software that can be used to write program for Kinect is Microsoft Visual Studio. This software can work together with Kinect for Windows SDK to perform coding and debugging any program for Kinect. Visual Studio 2012 has several language which are VB or Visual Basic ,C#, C and C++ [16]. In this project, Kinect GUI and program were built using VB language.

Enabling Kinect programming in Visual Studio requires a simple step. Figure 3.12 shows how to enable Kinect plug in in Visual Studio 2012.

```
Imports System
Imports System.IO.Ports

Imports System.Text
Imports Microsoft.Kinect
Imports Coding4Fun.Kinect.Wpf
```

Figure 3.12 Enabling Kinect in Visual Studio 2012

Since Visual Studio need to send data to Arduino Uno using serial communication function 'Imports System.IO.Ports' need to be enabled so Visual Studio have an access to send data to Arduino using serial communication through USB port connected to Arduino Uno board. Need to remind that Kinect for Windows SDK need to be installed first before the function to enable Kinect can be used.

Hand gesture will be used as motion to control the direction of output ball and also type of spin. In order to do that the movement of left hand and right hand captured by Kinect need to be scaled up from 0 until 180. This is because both of servo motor only can rotate up to 180°. Figure shows how the conversation process is done in Visual Basic 2012

```
Private Sub ScalePosition(ByVal element As FrameworkElement, ByVal joint As Joint, ByVal JID As JointType)
    'convert the value to X/Y
    'convert & scale (.3 = means 1/3 of joint distance)
    Dim scaledJoint As Joint = joint.ScaleTo(180, 180, 0.3F, 0.3F)

    Canvas.SetLeft(element, scaledJoint.Position.X)
    Canvas.SetTop(element, scaledJoint.Position.Y)
End Sub
```

Figure 3.13 Scaling value X and Y axis for all Joint captured by Kinect.

After scaling the value of X and Y axis, all of the value need to be sent to Arduino board using serial communication. To setup the serial communication we need a set of function dedicated for the task. Figure 3.14 shows how to setup serial communication in Visual Studio using Visual Basic language. Line 1 until line 9 is

the syntax code for setting up serial communication and line 2 is the syntax for selecting USB port that has been connected to Arduino Uno.

```
Private Sub ArduinoSetSerial()
    'Dim ArduinoCom As String = ComPort.Text
1   _serialPort = New SerialPort()
2   _serialPort.PortName = "COM11"
3   _serialPort.BaudRate = 9600
4   _serialPort.Parity = 0
5   _serialPort.DataBits = 8
6   _serialPort.StopBits = 1
7   _serialPort.Handshake = 0
8   _serialPort.ReadTimeout = 500
9   _serialPort.WriteTimeout = 500

    ComPort.Text = _serialPort.PortName

End Sub
```

Figure 3.14 Setting serial communication in Visual Studio 2012

In order to differentiate between left hand and right hand, a set of value need to be assigned to each joint detected by Kinect Skeletal system. Kinect for Windows SDK already given specific code for each of joint detected by Kinect. This code then can be used to differentiate each joint and make the programming easier at Arduino IDE.

```

Private Sub SendToArduino(ByVal joint As Joint, ByVal JID As JointType)

    Dim scaledJoint = joint.ScaleTo(180, 180, 0.3F, 0.3F)

    If JID = JointType.HandRight Then
        HRKinectX.Text = scaledJoint.Position.X
        HRKinectY.Text = scaledJoint.Position.Y
    End If

    If JID = JointType.HandLeft Then
        HLKinectX.Text = scaledJoint.Position.X
        HLKinectY1.Text = scaledJoint.Position.Y
    End If

    If JID = JointType.ElbowLeft Then
        ELKinectX.Text = scaledJoint.Position.X
        ELKinectY.Text = scaledJoint.Position.Y
    End If

    If JID = JointType.Head Then
        headX.Text = scaledJoint.Position.X
        headY.Text = scaledJoint.Position.Y
    End If

    If JID = JointType.ShoulderRight Then
        leftshoulderX.Text = scaledJoint.Position.X
        leftshoulderY.Text = scaledJoint.Position.Y
    End If

    If JID = JointType.ElbowRight Then
        rightelbowX.Text = scaledJoint.Position.X
        rightelbowY.Text = scaledJoint.Position.Y
    End If

```

Figure 3.15 Coding section for requesting predefined code for each joint.

```

ArduinoSendByte(scaledJoint.Position.X, scaledJoint.Position.Y, 1, JID)

```

Figure 3.16 Function to send X value, Y value and joint code to Arduino Uno board.

Figure 3.16 shows how the value of X and Y that has been scaled up will be sent to Arduino board together with joint code so the Arduino will have information on which servo required changes in value based on the hand gesture provided by user. In simple word which hand is moving now, either right hand or left hand.

3.9 Summary

In this chapter, the methodology of project was described clearly for each stage. Hence a proper planning project process flow was discussed briefly on this chapter and followed by component description together with circuit design of power supply system. Each main programming code of control system using Kinect has been described with figure shown.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

In overall, the development progress of the Kinect based Table Tennis Trainer System was considered quite smooth but a slight major deviate from schedule. The power supply circuit was successfully constructed and worked. Meanwhile, the programming of system was achieve a satisfactory level but need further development and tweaking performance to achieve fully functional and implementation stage.

4.2 Power Supply Circuit

The power supply circuit of Table tennis trainer was successfully constructed and function well with a little tweaking and troubleshooting. Figure 4.1 shows the layout of power supply circuit. Another 6V voltage regulator has been added in this circuit since the current drive from one 6V voltage regulator is not enough for two

servo motors. The 3.3 voltage regulator also has been changed to 5V voltage regulator to make it more safety and delivered more stable power to DC motor. Figure 4.2 shows the position of AC and DC connection for switching power supply 12V 4.5A. The output will be DC connected to the power supply circuit while the input will be AC connection.

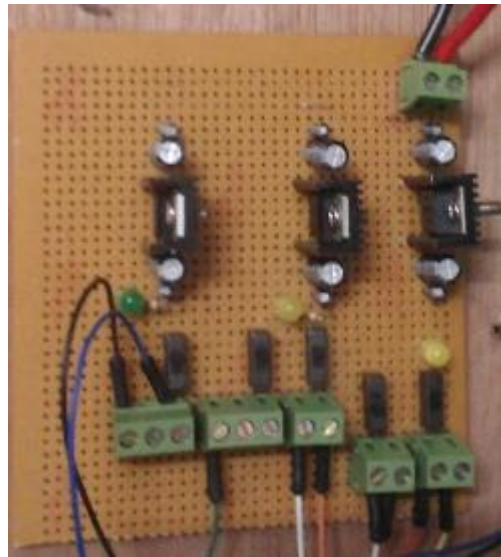


Figure 4.1 Layout of power supply circuit.

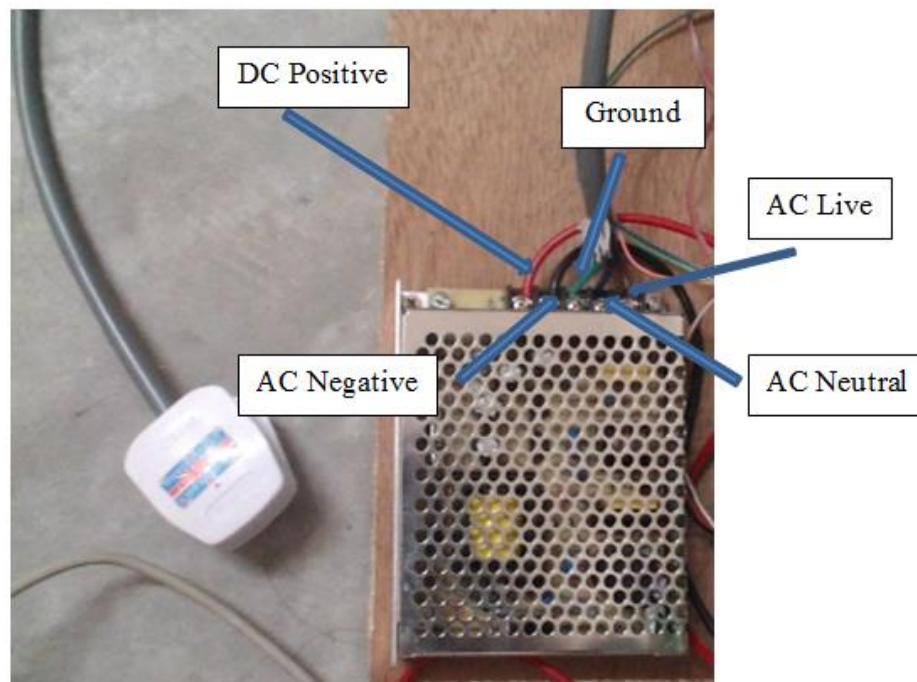


Figure 4.2 Switching power supply 12V 4.5A

4.3 Kinect Graphical User Interface

The Kinect Graphical User Interface for table tennis Trainer was designed using Microsoft Visual Studio 2012. User can change the elevation angle for Kinect. In addition, start and stop button also were prepared in case of any accident or emergency happen. The GUI also has X and Y axis meter which indicate the current value for X and Y axis for each joint. Figure 4.3 shows the GUI for Kinect based Table Tennis Trainer system.

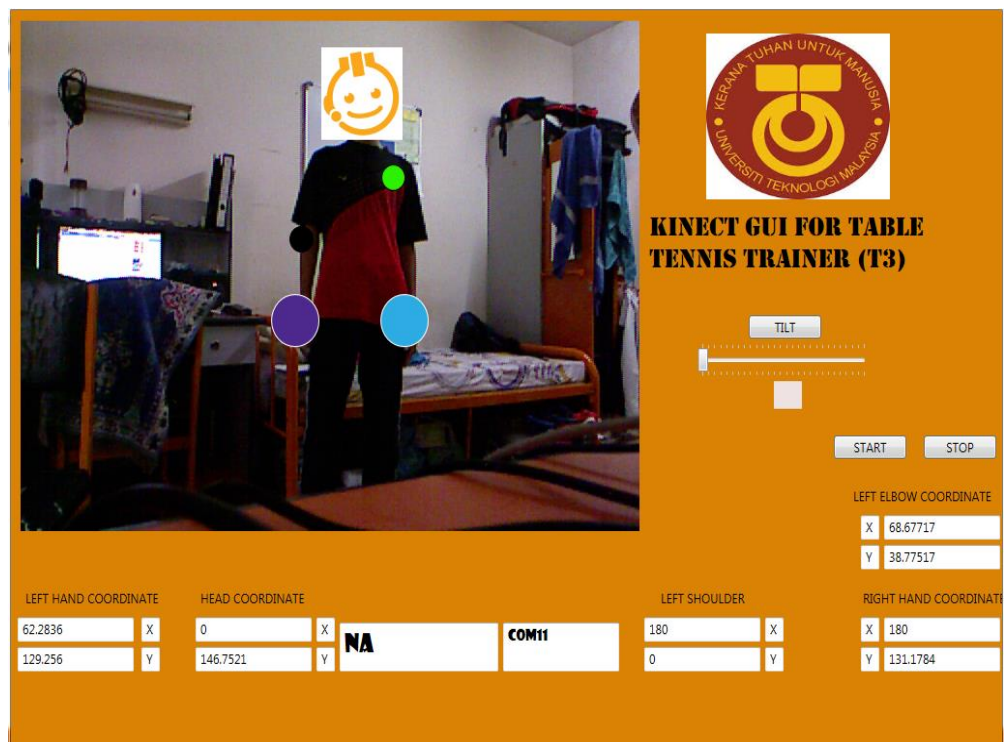


Figure 4.3 Graphical User Interface for Kinect based Table Tennis Trainer System

4.4 System Performance

The system was design to allow the user to select the type of spin and direction of the output ball. The type of spin depends on the spinner which functions to rotate the shooter head by using servo motor. This spinner will be controlled by right hand gesture. Figure 4.4 and 4.5 show the different type of spin controlled by hand gesture.



Figure 4.4 Right hand gesture for side spin



Figure 4.5 Right hand gesture for top spin

Meanwhile for the direction of the ball, player needs to use left hand to change to left or right. It is controlled by yaw which uses servo motor too as the actuator to move to right or left. Figure 4.6 and 4.7 show how the gesture controls the yaw part of 3T.



Figure 4.6 Left hand gesture for right side output



Figure 4.7 Left hand gesture for left side output

In order to enable control mode, player need to change the status of Kinect from 'NA' or not available to 'OK' state. This function works for as lock state since Kinect will continuously detect player movement and hand gesture even after player already chose their desired preference. If 'NA' status appears, Kinect will not send any signal to Arduino Uno and player can resume their training session without any interruptions. Figure 4.8 shows how to change 'NA' status to 'OK' status.



Figure 4.8 OK status gesture

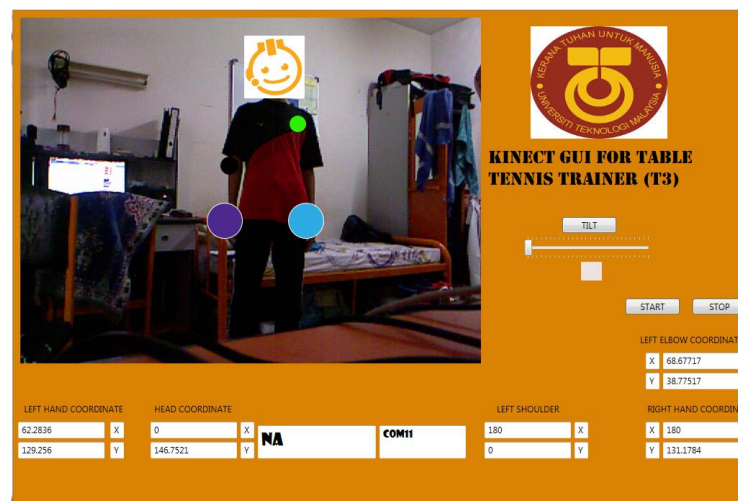


Figure 4.9 NA status gesture

The lock gesture used left elbow instead of hand to be detected by Kinect as the joint type for locking mechanism. In term of accuracy of this system, it depends on the distance between player and Kinect. The farther the distance of player with Kinect the less accuracy this system will operate. If the player stand too close to the Kinect, The system will not be able to detect any gesture come from player. In term of speed, this system manages to send data to Arduino Uno in short time with the help of laptop in processing the input data.

4.5 Summary

As a summary for Chapter 4, the result of Kinect based Table Tennis Trainer was satisfaction but need some improvement in coding part for Kinect. Meanwhile for the power supply circuit the performance was great with capability to continuously supply electrical power to 3T without failure.

CHAPTER 5

PROJECT MANAGEMENT

5.1 Introduction

Project management is essential to achieve all project goal and objective by effective project planning, organizing and controlling resources within specific period time. A successful project not only measure from its performance but also need to take account the time and cost of project whether it is worth or not. Figure 5.1 shows the time cost and quality triangle in project management [21].

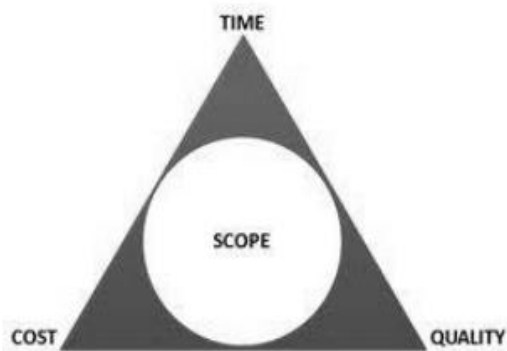


Figure 5.1 Project Management – Time, Cost, quality triangle

A clear and firm project scope has been target in the eraly of project with consideration of time and cost of project for quality control. The primary constraints in this project are the research time, research budget, research scope and human resources to complete every scheduled task. As an alternative for these constraints, project schedule been tabulated in form of Gantt chart for a clear vision or act as a guideline in time management of project. Next, cost estimation is very important in budget controlling which can ensure the project cost will not go beyond the limit while keeping the project until the end of Gantt chart. Performing market survey on components required and their suppliers and information from the survey need to be tabulated to obtain the final cost for the project.

5.2 Project Schedule

Figure 5.2 shows the project Gantt chart for semester one. This project expected will be started in September 2014 and the actual schedule starts during second week of September 2014 after one week student return from their industrial training. At the early stage of project, it was essential to identify the problem for research tittle finding. Next every skill required for the next part of final year project such as Arduino programming and building GUI will be learned until December 2014.

	September(2014)				October(2014)				November(2014)				December(2014)			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
Task																
Problem Identification																
Discussion with SV																
Literature review																
Learn coding Arduino																
Learn build GUI using Microsoft VB																
Learning Kinect																
Project Presentation Preparation(FYP1)																
Project Presentation(FYP1)																
Project Report(FYP1)																

Figure 5.2 Project Gantt chart (semester one)

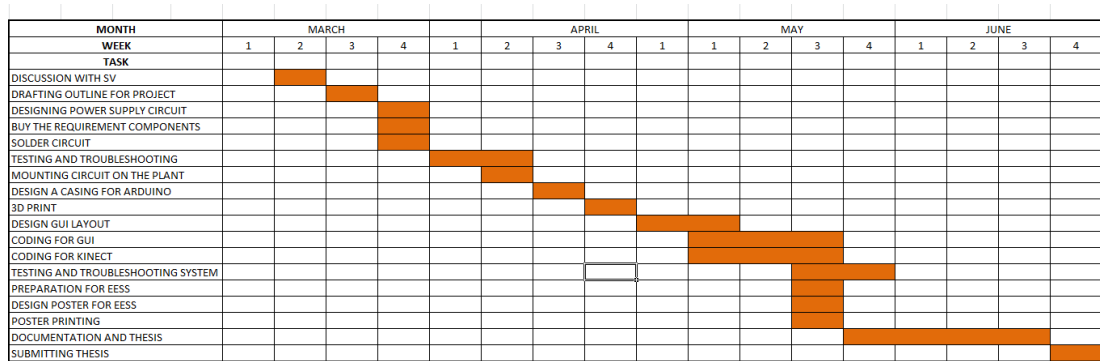


Figure 5.3 Project Gantt chart (semester two)

Table 5.3 shows project Gantt chart for semester two. Compare to semester one, there have slight different in term of delay taken is some task. The delay of project is due to late acquisition for Kinect hardware from CSI or Centre of Student Innovation. As Kinect is our main focus for this project, controller programming cannot proceed and causing the project delay and postpone.

5.3 Cost Estimation

Cost estimation for this project mainly for the construction of power supply circuit, microcontroller and motor driver. Since the hardware part and also Kinect can be borrowed by Dr. Zaharudin and CSI respectively, the cost for both of them in will not be counted in this estimation. Table 5.1 shows the cost estimation of assembling the power supply circuit after troubleshooting process.

Table 5-1 Cost Estimation for Electrical Hardware

Component name	Price	Unit	Subtotal
Arduino Uno R3 Compatible	RM58.00	1	RM58.00
2Amp Motor Driver Shield	RM44.52	1	RM44.52
Cytron 10A Motor Driver Shield	RM54.59	1	RM54.59
Electrolytic Capacitor 16v 330 μ F	RM0.40	6	RM2.40
Electrolytic Capacitor 16v 10 μ F	RM0.20	6	RM1.20
Voltage Regulator 3.3V	RM2.00	1	RM2.00
Voltage Regulator 6V	RM1.00	2	RM2.00
Voltage Regulator 5V	RM1.06	1	RM1.06
Donut Board 10x24cm	RM3.80	1	RM3.80
Switching Power Supply 12V 4.2A	RM66.50	1	RM66.50
Slide Switch 3 pins Black	RM1.00	4	RM4.00
Heat Sink 20x15x10 Black	RM0.70	3	RM2.10
Terminal Block DG128V-02 (GREEN)	RM0.70	6	RM4.20
Terminal Block DG128V-03 (GREEN)	RM1.00	4	RM4.00
Multicore Wire AWG14 (1m)	RM4.00	2	RM8.00
Subtotal			RM258.37

5.4 Summary

As summary for this chapter, Gantt chart has been prepared in early stage to ensure this project can be conduct smoothly and able to reschedule and remark if any delaying in schedule. Besides that, cost estimation and market survey of material was used to control the budget of project to ensure the budget did not over the limit of financial capability

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 Introduction

A lot of research using Kinect as the focus point has been conducted around the world. Until today, there is more than 10 paper work has been released for research about Kinect sensor in various type of field. As discussed in Chapter 2 all of the research mostly in image processing and medical field.

6.2 Conclusion

Kinect based Table Tennis Trainer system project has successfully met the objective and project scope stated in Chapter 1. In this project, a system that able to control 3T using Kinect to perform various type of spin and change the direction of the ball. The microcontroller module used in this project was Arduino Uno with Arduino IDE and Microsoft Visual Studio 2012 as joint platform. The power supply circuit was constructed according to design in the methodology of the Chapter 3 but have some minor changes to tweak its performance. The implementation of

algorithm to convert and scale X and Y axis for the joint type of Kinect to control the angle of the Servo motors.

6.3 Recommendations on Future Work

There are some limitations of the Kinect and 3T in this project. Firstly, the Kinect system itself has fluctuated reading of X and Y axis bring in result of unstable control of servo motors. Kinect system needs a proper and high level coding to make the Kinect as a proper and ready controller for 3T. The coding need to be optimized so that the fluctuated reading could be eliminated and more stable control of servo motor can be gained. Secondly since Kinect is very sensitive to motion, proper mechanism of 'lock' gesture needs to be developed so that the player can continue their training without Kinect interrupting their session. In this project the lock gesture has been made but not so efficient that sometimes if the user move around while practicing the Kinect will detect the movement and change the behavior of 3T.

In addition, the performance for the spin ball also slightly not so accurate. The hardware need to be optimized especially for the ball loading part and also for the shooter part. The loading part sometimes failed to load the ball properly into the ball shooter part. It can be optimized by changing the mechanism of the loading part. For the shooter part it might need a better mechanism of shooting since current mechanism not adequate to produce the type of spin stated by the previous developer.

Lastly, if the coding can be improved so that the Kinect can have response to shoot the ball according to player response, the system might give more excitement to player. This type of coding might be hard and need more expert skill but if can be achieved it will improve the system into the next level. With implementation of the recommendations and the modifications stated above, the Kinect based Table Tennis Trainer system with better performance and essential features can be expected.

REFERENCES

- 1 Cruz, Lucio and Velho (2012). Kinect and RGBD Images : Challenges and Applications. Graphics, Patterns and Images Tutorials (SIBGRAPI-T), 2012 25th SIBGRAPI Conference. 22-25 August. Ouro Preto: IEEE. 36 – 49.

- 2 Dobra, Maxim and Ceparu (2013). A Face Recognition System Based on a Kinect Sensor and Windows Azure Cloud Technology. Signals, Circuits and Systems (ISSCS), 2013 International Symposium. 11 – 12 July. Lasi: IEEE. 1 – 4.

- 3 Tahavori, Alnowami, Jones, Elangovan, Donovan, Wells (2013). Assessment of Microsoft Kinect Technology (Kinect for Xbox and Kinect for windows) For Patient Monitoring During External Beam Radiotherapy. Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2013 IEEE. 27 Oct. – 2 Nov. Seoul: IEEE. 1 - 5

- 4 El-laithy, Jidong Huang, Yeh (2012). Study on The Use of Microsoft Kinect for Robotics Applications. Position Location and Navigation Symposium (PLANS), 2012 IEEE/ION. 23 – 26 April. Myrtle Beach, SC: IEEE. 1280 – 1288.

- 5 NiDEX, “50W Dual Output Switching Power Supply”, D-50A datasheet, Nov. 2014.

- 6 NiDEX, “50W Single Output Switching Power Supply”, NES – 50 Series, Nov. 2014

- 7 Augarten, Stan (1983). The Most Widely Used Computer on a Chip: TMS 1000. In: Augarten. State of the Art: A Photographic History of the Integrated Circuit. New Haven and New York: Ticknor & Fields. 2; 1983
- 8 Texas Instrument Inc.” MSP-EXP430G2 LaunchPad Evaluation Kit User's Guide”,2010
- 9 K. Khoshelham (2011). Accuracy Analysis of Kinect Depth Data.ISPRS Calgary 2011 Workshop. 29-31 August. Calgary,Canada: ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences,133-138.
- 10 Monir, Rubya, Ferdous (2012). Rotation and Scale Invariant Posture Recognition using Microsoft Kinect Skeletal Tracking Feature. Intelligent Systems Design and Applications (ISDA), 2012 12th International Conference on. Kochi:IEEE,404-409
- 11 Cytron Technologies,”Shield-MD10 User’s Manual,” Product Specification and Limitations, Sept. 2014.
- 12 Cytron Technologies,”Shield-2AMotor User’s Manual,” Product Specification and Limitations, Feb 2015
- 13 “Description of Pololu 150:1 Metal Gearmotor,” 2013. [Online] Available: www.pololu.com.
- 14 Cytron Technologies, “RC Servo User Manual,” RC Servo C36R, C4R, C55R datasheet, Apr. 2013.
- 15 P. D. Minns, C Programming For the PC the MAC and the Arduino Microcontroller System. Author House, 2013.

- 16 Yongdong Wu (2009). Enhancing Security Check in Visual Studio C/C++ Compiler. Software Engineering, 2009. WCSE '09. WRI World Congress. 19- 21 Aug. Xiamen :IEEE, 109-113
- 17 Microsoft Corporation."Xbox 360 Kinect Sensor User's Manual,"2010.
- 18 Microsoft Corporation."Xbox 360 With Kinect Sensor Quick Setup Guide," Retrieved June 2015.
- 19 Brian W. Evans (2007). Arduino Programming Notebook. San Fransisco, California: Creative Commons.2007
- 20 Lee Poh Yong. Table Tennis Trainer (3T). degree of Bachelor Thesis. Universitit Teknologi Malaysia; 2014
- 21 M. a. Mantel, Project Management: A Managerial Approach, 2010.

APPENDIX A

Doing EESS exhibition 2015 MB04-07 with supervisor.



APPENDIX B

1) Microsoft Visual Basic:

```
Imports System
Imports System.IO.Ports
```

```
Imports System.Text
Imports Microsoft.Kinect
Imports Coding4Fun.Kinect.Wpf
```

```
Namespace SkeletalTracking
```

```
    "" <summary>
    "" Interaction logic for MainWindow.xaml
    "" </summary>
```

```
    Partial Public Class MainWindow
        Inherits Window
```

```
        'Arduino start
```

```
        Public Shared _continue As Boolean
        Public Shared _serialPort As SerialPort
        Public Shared ScreenMaxX As Integer = 180
        Public Shared ScreenMaxY As Integer = 180
        'Arduino end
```

```
        Public Sub New()
            InitializeComponent()
        End Sub
```

```
        Private _closing As Boolean = False
        Private Const skeletonCount As Integer = 6
        Private allSkeletons(skeletonCount - 1) As Skeleton
```

```
        Private Sub Window_Loaded(ByVal sender As Object, ByVal e As
RoutedEventArgs)
            AddHandler kinectSensorChooser1.KinectSensorChanged, AddressOf
kinectSensorChooser1_KinectSensorChanged
```

```
        'arduino start
```

```

    ArduinoSetSerial()
    ArduinoOpenSerial()
'arduino end

```

```
End Sub
```

```

    Private Sub KinectSensorChooser1_KinectSensorChanged(ByVal sender
As Object, ByVal e As DependencyPropertyChangedEventArgs)
        Dim old As KinectSensor = CType(e.OldValue, KinectSensor)

```

```
        StopKinect(old)
```

```
        Dim sensor As KinectSensor = CType(e.NewValue, KinectSensor)
```

```
        If sensor Is Nothing Then
```

```
            Return
```

```
        End If
```

```

        Dim parameters = New TransformSmoothParameters With { .Smoothing
= 0.3F, .Correction = 0.0F, .Prediction = 0.0F, .JitterRadius = 1.0F,
        .MaxDeviationRadius = 0.5F }
        sensor.SkeletonStream.Enable(parameters)

```

```
        sensor.SkeletonStream.Enable()
```

```

        AddHandler sensor.AllFramesReady, AddressOf
        sensor_AllFramesReady

```

```
        sensor.DepthStream.Enable(DepthImageFormat.Resolution640x480Fps30)
```

```
        sensor.ColorStream.Enable(ColorImageFormat.RgbResolution640x480Fps30)
```

```
        Try
```

```
            sensor.Start()
```

```
        Catch e1 As System.IO.IOException
```

```
            KinectSensorChooser1.AppConflictOccurred()
```

```
        End Try
```

```
    End Sub
```

```

    Private Sub sensor_AllFramesReady(ByVal sender As Object, ByVal e As
AllFramesReadyEventArgs)

```

```

If _closing Then
    Return
End If

```

```

'Get a skeleton

```

```

Dim first As Skeleton = GetFirstSkeleton(e)

```

```

If Not first Is Nothing Then

```

```

    'set scaled position

```

```

        ScalePosition(headImage, first.Joints(JointType.Head),
JointType.Head)
        ScalePosition(leftEllipse, first.Joints(JointType.HandLeft),
JointType.HandLeft)
        ScalePosition(rightEllipse, first.Joints(JointType.HandRight),
JointType.HandRight)
        ScalePosition(elbowleft, first.Joints(JointType.ElbowLeft),
JointType.ElbowLeft)
        ScalePosition(leftshoulder, first.Joints(JointType.ShoulderRight),
JointType.ShoulderRight)
        ScalePosition(rightelbow, first.Joints(JointType.ElbowRight),
JointType.ElbowRight)

```

```

        GetCameraPoint(first, e)

```

```

        SendToArduino(first.Joints(JointType.HandLeft),
JointType.HandLeft)
        SendToArduino(first.Joints(JointType.HandRight),
JointType.HandRight)
        SendToArduino(first.Joints(JointType.ElbowLeft),
JointType.ElbowLeft)
        SendToArduino(first.Joints(JointType.Head), JointType.Head)
        SendToArduino(first.Joints(JointType.ShoulderRight),
JointType.ShoulderRight)
        SendToArduino(first.Joints(JointType.ElbowRight),
JointType.ElbowRight)

```

```

    End If

```

```

End Sub

```

```

Private Sub GetCameraPoint(ByVal first As Skeleton, ByVal e As
AllFramesReadyEventArgs)

```

```

    Using depth As DepthImageFrame = e.OpenDepthImageFrame()
        If depth Is Nothing OrElse kinectSensorChooser1.Kinect Is Nothing
Then
            Return
        End If
    End Using

```

'Map a joint location to a point on the depth map

'head

```
Dim headDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.Head).Position)
'left hand
Dim leftDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.HandLeft).Position)
'right hand
Dim rightDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.HandRight).Position)
'left elbow
Dim leftelbowDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.ElbowLeft).Position)
'left shoulder
Dim rightshoulderDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.ShoulderRight).Position)
'right elbow
Dim rightelbowDepthPoint As DepthImagePoint =
depth.MapFromSkeletonPoint(first.Joints(JointType.ElbowRight).Position)
```

'Map a depth point to a point on the color image

'head

```
Dim headColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(headDepthPoint.X, headDepthPoint.Y,
ColorImageFormat.RgbResolution640x480Fps30)
'left hand
Dim leftColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(leftDepthPoint.X, leftDepthPoint.Y,
ColorImageFormat.RgbResolution640x480Fps30)
'right hand
Dim rightColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(rightDepthPoint.X, rightDepthPoint.Y,
ColorImageFormat.RgbResolution640x480Fps30)
'left elbow
Dim leftelbowColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(leftelbowDepthPoint.X, leftelbowDepthPoint.Y,
ColorImageFormat.RgbResolution640x480Fps30)
'left shoulder
Dim rightshoulderColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(rightshoulderDepthPoint.X,
rightshoulderDepthPoint.Y, ColorImageFormat.RgbResolution640x480Fps30)
'right elbow
Dim rightelbowColorPoint As ColorImagePoint =
depth.MapToColorImagePoint(rightelbowDepthPoint.X,
rightelbowDepthPoint.Y, ColorImageFormat.RgbResolution640x480Fps30)
```

'Set location

```
CameraPosition(headImage, headColorPoint)
```

```

CameraPosition(leftEllipse, leftColorPoint)
CameraPosition(rightEllipse, rightColorPoint)
CameraPosition(elbowleft, leftelbowColorPoint)
CameraPosition(leftshoulder, rightshoulderColorPoint)
CameraPosition(rightelbow, rightelbowColorPoint)

```

```

End Using
End Sub

```

```

Private Function GetFirstSkeleton(ByVal e As AllFramesReadyEventArgs)
As Skeleton

```

```

    Using skeletonFrameData As SkeletonFrame = e.OpenSkeletonFrame()
        If skeletonFrameData Is Nothing Then
            Return Nothing
        End If

```

```

        skeletonFrameData.CopySkeletonDataTo(allSkeletons)

```

```

        'get the first tracked skeleton

```

```

        Dim first As Skeleton = ( _
            From s In allSkeletons _
            Where s.TrackingState = SkeletonTrackingState.Tracked _
            Select s).FirstOrDefault()

```

```

        Return first

```

```

    End Using
End Function

```

```

Private Sub StopKinect(ByVal sensor As KinectSensor)

```

```

    If sensor IsNot Nothing Then
        If sensor.IsRunning Then
            'stop sensor
            sensor.Stop()

```

```

            'stop audio if not null

```

```

            If sensor.AudioSource IsNot Nothing Then
                sensor.AudioSource.Stop()
            End If

```

```

        End If
    End If
End Sub

```

```

Private Sub CameraPosition(ByVal element As FrameworkElement, ByVal
point As ColorImagePoint)

```

```

    'Divide by 2 for width and height so point is right in the middle
    ' instead of in top/left corner

```

```

    Canvas.SetLeft(element, point.X - (element.Width \ 2))

```

```
Canvas.SetTop(element, point.Y - element.Height \ 2)
```

```
End Sub
```

```
Private Sub ScalePosition(ByVal element As FrameworkElement, ByVal  
joint As Joint, ByVal JID As JointType)
```

```
    'convert the value to X/Y
```

```
    'convert & scale (.3 = means 1/3 of joint distance)
```

```
    Dim scaledJoint As Joint = joint.ScaleTo(180, 180, 0.3F, 0.3F)
```

```
    Canvas.SetLeft(element, scaledJoint.Position.X)
```

```
    Canvas.SetTop(element, scaledJoint.Position.Y)
```

```
End Sub
```

```
Private Sub SendToArduino(ByVal joint As Joint, ByVal JID As  
JointType)
```

```
    Dim scaledJoint = joint.ScaleTo(180, 180, 0.3F, 0.3F)
```

```
    If JID = JointType.HandRight Then
```

```
        HRKinectX.Text = scaledJoint.Position.X
```

```
        HRKinectY.Text = scaledJoint.Position.Y
```

```
    End If
```

```
    If JID = JointType.HandLeft Then
```

```
        HLKinectX.Text = scaledJoint.Position.X
```

```
        HLKinectY1.Text = scaledJoint.Position.Y
```

```
    End If
```

```
    If JID = JointType.ElbowLeft Then
```

```
        ELKinectX.Text = scaledJoint.Position.X
```

```
        ELKinectY.Text = scaledJoint.Position.Y
```

```
    End If
```

```
    If JID = JointType.Head Then
```

```
        headX.Text = scaledJoint.Position.X
```

```
        headY.Text = scaledJoint.Position.Y
```

```
    End If
```

```
    If JID = JointType.ShoulderRight Then
```

```
        leftshoulderX.Text = scaledJoint.Position.X
```

```
        leftshoulderY.Text = scaledJoint.Position.Y
```

```
    End If
```

```

If JID = JointType.ElbowRight Then
    rightelbowX.Text = scaledJoint.Position.X
    rightelbowY.Text = scaledJoint.Position.Y

```

```

End If

```

```

    ArduinoSendByte(scaledJoint.Position.X, scaledJoint.Position.Y, 1, JID)

```

```

End Sub

```

```

Private Sub ArduinoSendByte(ByVal kinect_x As Single, ByVal kinect_y
As Single, ByVal kinect_z As Single, ByVal kinect_j As Integer)

```

```

    Dim x, y, z, j As Byte
    Dim sx, sy As Single
    Dim HowOften As Integer
    ComStatus.Text = "NA"

```

```

    x = Math.Abs(CByte(kinect_x))
    y = Math.Abs(CByte(kinect_y))
    z = CByte(kinect_z)
    j = CByte(kinect_j)

```

```

    x = x

```

```

    Dim ArduinoBuffer() As Byte = {x, y, z, j}
    If _serialPort.IsOpen Then
        ComStatus.Text = "OK"
        _serialPort.Write(ArduinoBuffer, 0, ArduinoBuffer.Length)
    End If

```

```

    If j = 5 And y > 0 Then

```

```

        ArduinoCloseSerial()

```

```

    End If

```

```

    If j = 5 And y = 0 Then

```

```

        ArduinoOpenSerial()

```

```

    End If

```

```

End Sub

```

```
Private Sub Window_Closing(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs)
```

```
    _closing = True
    StopKinect(kinectSensorChooser1.Kinect)
```

```
End Sub
```

```
Private Sub ArduinoSetSerial()
    _serialPort = New SerialPort()
    _serialPort.PortName = "COM11"
    _serialPort.BaudRate = 9600
    _serialPort.Parity = 0
    _serialPort.DataBits = 8
    _serialPort.StopBits = 1
    _serialPort.Handshake = 0
    _serialPort.ReadTimeout = 500
    _serialPort.WriteTimeout = 500
```

```
    ComPort.Text = _serialPort.PortName
```

```
End Sub
```

```
Private Sub ArduinoOpenSerial()
    If Not _serialPort.IsOpen Then
        _serialPort.Open()
```

```
        ComStatus.Text = "OK"
```

```
    End If
End Sub
```

```
Private Sub ArduinoCloseSerial()
    If _serialPort.IsOpen Then
        _serialPort.Close()

        ComStatus.Text = "DISCONNECT"
    End If
End Sub
```

```
Private Sub stopbutton_Click(ByVal sender As Object, ByVal e As
RoutedEventArgs) Handles stopbutton.Click
```

```
    ArduinoOpenSerial()
```

```
    ArduinoSendByte(0, 0, 1, 30)
```

```

        ComStatus.Text = "RESET"

        ArduinoCloseSerial()

    End Sub

    Private Sub tilt_Click(ByVal sender As Object, ByVal e As
RoutedEventArgs) Handles tilt.Click

        tilt.IsEnabled = False

        If kinectSensorChooser1.Kinect IsNot Nothing AndAlso
kinectSensorChooser1.Kinect.IsRunning Then
            kinectSensorChooser1.Kinect.ElevationAngle =
CInt(Fix(tiltslider.Value))
            labelcurrentangle.Content =
kinectSensorChooser1.Kinect.ElevationAngle

        End If

        System.Threading.Thread.Sleep(New TimeSpan(hours:=0, minutes:=0,
seconds:=1))
        tilt.IsEnabled = True

    End Sub

    Private Sub Slider_ValueChanged_1(sender As Object, e As
RoutedPropertyChangedEventArgs(Of Double))

    End Sub

    Private Sub startbutton_Click(ByVal sender As Object, ByVal e As
RoutedEventArgs) Handles startbutton.Click

        ArduinoOpenSerial()

        ArduinoSendByte(0, 0, 1, 31)

        ComStatus.Text = "OK"

    End Sub
End Class
End Namespace

```

2) Arduino part:

```

#include <Servo.h>

int MotorDC_PWMR = 5; //initialization of pin for DC motor
int MotorDC_PWML = 6;
int MotorDC_PWMBR = 3;

Servo yaw servo;    //global declaration for two servo
Servo pitch servo;

void setup()
{
  // start serial port at 9600 bps:
  Serial.begin(9600);

  yaw servo.attach(9);    //initialization pin for yaw servo
  pitch servo.attach(8);  //initialization pin for pitch servo
  yaw servo.write(90);    //default position
  pitch servo.write(0);   //default position

  //declaration for all of the pin for DC motor part
  pinMode(MotorDC_PWMBR,OUTPUT);
  pinMode(MotorDC_PWML,OUTPUT);
  pinMode(MotorDC_PWMR,OUTPUT);
  pinMode(7,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(2,OUTPUT);
  pinMode(9,OUTPUT);

  //setting direction of the rotation for the DC motor
  digitalWrite(4,LOW);
  digitalWrite(7,LOW);
  digitalWrite(2,HIGH);

}

unsigned char x,y,z,j =0;
int val = 0;

void loop()
{

  // read the 4-byte byffer sent from VB containing coordinates and joint ID.
  if (Serial.available() >= 4) {

```

```

    analogWrite(MotorDC_PWMBR,100);    // turn on the DC motor once the
head has been detected
    analogWrite(MotorDC_PWMR,255);
    analogWrite(MotorDC_PWML,255);

    x = Serial.read();
    y = Serial.read();
    z = Serial.read();
    j = Serial.read();

    if (j == 7) { // HandLeft
        yaw servo.write(180-((9/4)*y));    // sets the servo position according
to the scaled value
        delay(10);                        // waits for the servo to get there
    }

    if (j == 11) { // HandRight
        pitch servo.write(y-94);           // sets the servo position according to the
scaled value
        delay(10);                        // waits for the servo to get there
    }

    if(j == 30)
    {
        // turn off all DC motor once stop button command has been received
        analogWrite(MotorDC_PWMBR,0);
        analogWrite(MotorDC_PWMR,0);
        analogWrite(MotorDC_PWML,0);

        //return back all servo motor into default position
        yaw servo.write(90);
        pitch servo.write(0);
    }

    //turn on all DC motor once Start button command has been received
    if (j == 31){
        analogWrite(MotorDC_PWMBR,100);
        analogWrite(MotorDC_PWMR,255);
        analogWrite(MotorDC_PWML,255);

    }

}

}
}

```